Phosphite : Incitation à la collaboration pour la vidéo à la demande en P2P

Mary-Luc Champel, Anne-Marie Kermarrec, Nicolas Le Scouarnec

Phosphite : Incitation à la collaboration pour la vidéo à la demande en P2P

Mary-Luc Champel[1], Anne-Marie Kermarrec[2] et Nicolas Le Scouarnec[1]

[1] *Thomson R&D France , 1 avenue de Belle-Fontaine, 35576, Cesson-Sevigné, France*
[2] *INRIA Rennes / Bretagne-Atlantique, campus de Beaulieu, 35042, Rennes, France*

Nous proposons *Phosphite*, un mécanisme d'incitation qui vise à assurer que les pairs d'un système de vidéo à la demande dédient une partie de leur bande passante à du téléchargement dans le désordre. Sans incitation, les pairs égoïstes vont tenter de maximiser leur bénéfice en téléchargeant le contenu dans l'ordre, ce qui va à l'encontre de l'intérêt général. Les derniers blocs téléchargeables ayant alors tendance à disparaître du système, le serveur devra souvent réintroduire ces derniers blocs. *Phosphite* permet de s'assurer que les pairs dédient une partie de leur bande passante à du téléchargement dans le désordre au bénéfice de la communauté. Ainsi, tous les blocs restent disponibles avec une probabilité supérieure à 98% alors que sans incitation les derniers blocs sont disponibles avec une probabilité inférieure à 50%.

Keywords: pair-à-pair, incitation, vidéo à la demande

1 Introduction

Dans les systèmes de vidéo à la demande, les utilisateurs choisissent un contenu et le regardent quand ils le souhaitent. Les utilisateurs ne sont pas synchronisés, ce qui rend impossible l'utilisation de multicast pour réduire les besoins en ressources au niveau du serveur. Ainsi, le déploiement de solutions P2P (pair à pair) limite la charge du serveur en permettant aux pairs d'échanger du contenu entre eux.

Les systèmes de vidéo à la demande s'appuyant sur des technologies P2P utilisent en général une variante du protocole *Bittorrent* [Coh03]. Le contenu à télécharger est divisé en blocs. La copie initiale des blocs est fournie par le serveur. Par la suite, les pairs, agissant à la fois comme clients et comme relais, échangent des blocs jusqu'à avoir obtenu l'ensemble du contenu. Ainsi, un pair ne s'adresse au serveur que si le bloc souhaité n'est pas disponible auprès d'un de ses voisins. Les recours au serveur sont coûteux et doivent rester exceptionnels.

Bittorrent [Coh03] est régi par quelques règles qui assurent son bon fonctionnement. D'une part, afin de favoriser la collaboration, les pairs sont servis proportionnellement à leur contribution via un mécanisme de réciprocation. D'autre part, afin d'assurer une diversité suffisante entre les pairs, le bloc qui doit être téléchargé est choisi selon une politique *LRF (Local Rarest First)*. Cette politique indique que le bloc à télécharger en priorité est celui qui est le plus rare localement (estimation réalisée selon les cartes de disponibilités fournies par les voisins). La diversité assurée par cette seconde règle est cruciale puisqu'elle permet le bon fonctionnement du mécanisme de réciprocation et qu'elle assure qu'aucun bloc ne devient rare, rendant le système résistant au dynamisme. En effet, l'absence de blocs rares permet de garantir que les pairs pourront télécharger l'ensemble des blocs même si un nombre important de pairs quitte le système.

Les systèmes de vidéo à la demande [PWMC08, CE07, CSDP07] qui s'appuient sur *Bittorrent* identifient la politique *LRF* comme une composante essentielle du protocole *Bittorrent*. Néanmoins, la politique *LRF* conduit à un téléchargement dans le désordre qui ne permet pas la réalisation d'un système de vidéo à la demande dans lequel l'utilisateur commence à lire le contenu alors qu'il n'en a téléchargé qu'une petite partie. Ainsi, la politique de choix des blocs utilisée dans de tels systèmes combine une part importante de téléchargement dans l'ordre à une petite part de téléchargement dans le désordre.

La nature distribuée des réseaux P2P les rend sensibles aux comportements égoïstes. En effet, chaque pair est indépendant et libre de modifier son comportement en s'appuyant sur une variante du protocole, ce qui

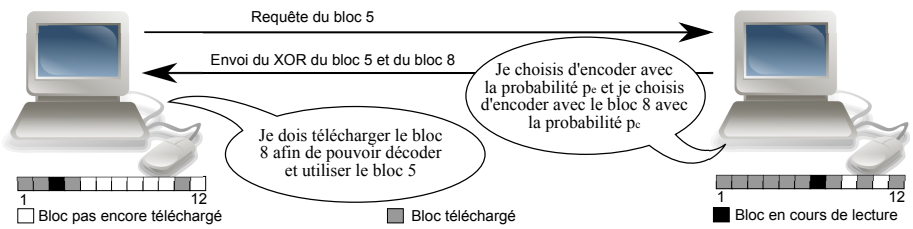


FIGURE 1: Illustration d'un échange entre deux pairs. Le second pair répond à la requête du premier tout en lui imposant de télécharger le bloc 8 avant de pouvoir utiliser le bloc 5.

aura un impact important sur le reste du système. Ainsi, un comportement égoïste apparaîtra dès lors que le pair arrête de servir l'intérêt général pour se concentrer sur son propre intérêt. À ce titre, *Bittorrent* inclut déjà un mécanisme de réciprocation pour s'assurer que les pairs collaborent bien en fournissant du contenu aux autre pairs. Cependant, aucun mécanisme ne s'assure que le pair effectue bien un téléchargement dans le désordre, *Bittorrent* reposant entièrement sur la bonne volonté des pairs pour ce point.

Nous nous intéressons au cas où les pairs téléchargent le contenu dans l'ordre. Dans un système de vidéo à la demande, chaque pair à intérêt à télécharger le contenu dans l'ordre puisqu'il téléchargera en premier le contenu qu'il utilisera en premier. Une telle stratégie permet de s'assurer que le contenu est disponible au moment de l'afficher à l'écran. Ainsi, les pairs ont individuellement intérêt à ne faire que du téléchargement dans l'ordre, ce qui est malheureusement contraire à l'intérêt général.

Les comportements égoïstes ont un impact réel sur les réseaux P2P. En effet, l'anonymat des utilisateurs ne les incite pas à contribuer. Ainsi, GNutella, un des premiers réseau P2P, comportait plus de 80% de *free-riders* qui utilisaient le système sans contribuer, ce qui dégradait les performances pour les utilisateurs légitimes. *Bittorrent* a supplanté ce réseau puisqu'il assure la contribution en récompensant,par de meilleures performances , les pairs qui contribuent. De plus, un comportement égoïste qui conduirait les pairs à télécharger dans l'ordre semble possible. En effet, afin de permettre aux utilisateurs de visualiser un aperçu du contenu en cours de téléchargement, certains clients *Bittorrent (Bittornado, G3 Torrent, uTorrent)* incluent une option, fortement déconseillée, permettant de télécharger les blocs dans l'ordre.

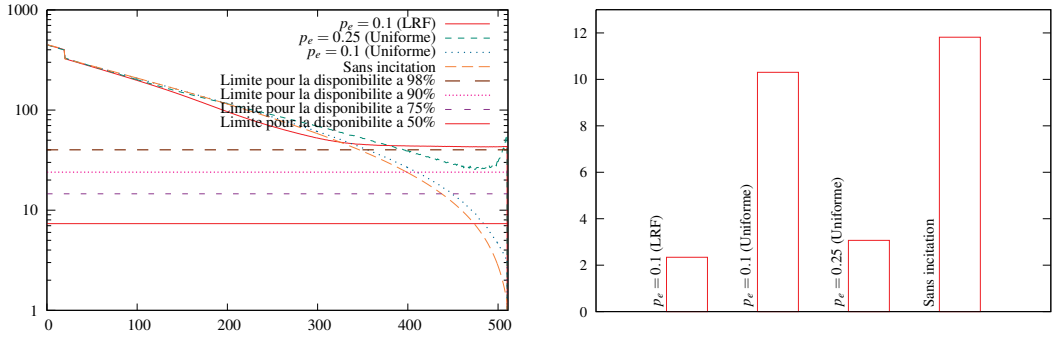
Phosphite, le mécanisme que nous proposons, permet de s'assurer qu'un pair dédie bien une partie (10%) de sa bande passante à du téléchargement dans le désordre. Le but est d'assurer que chaque pair participera à un tâche commune assurant la persistance des données. Nous nous concentrons sur le respect de l'ordre de téléchargement. En effet, la collaboration peut-être assurée par les mécanismes classiques de réciprocation déjà présents dans *Bittorrent*. Dans la suite de l'article, nous présentons *Phosphite* et nous abordons la manière de structurer les paquets pour assurer la robustesse du mécanisme. Enfin, nous évaluons l'effet du mécanisme d'incitation.

2 *Phosphite*, Préserver le téléchargement *Out-Of-Order*

Phosphite est une extension d'un protocole de vidéo à la demande s'inspirant de *Bittorrent*. Cette extension permet de s'assurer qu'un pair dédiera une partie de sa bande passante à du téléchargement dans le désordre. Cette obligation est imposée à un pair par les pairs qui lui fournissent des données. En effet, le pair qui demande à télécharger un premier bloc est parfois forcé d'en télécharger un second avant de pouvoir utiliser le premier. *Phosphite* ne s'appuie ni sur des protocoles d'audit, ni sur des mécanismes de cryptographie complexes à déployer. Au final, on répond au pair en lui envoyant le bloc mais en conditionnant son utilisation à la possession d'un autre bloc grâce à un petit challenge informatique.

Phosphite est relativement simple. Un pair qui reçoit une requête choisit d'imposer le téléchargement d'un autre bloc avec une probabilité $p_e(x)$. S'il a choisi d'imposer le téléchargement, il choisit un bloc $y > x$ selon une loi définie par $p_c(x,y)$. Le challenge est construit à partir du code correcteur d'erreurs $(x, y, x \oplus y)$. Le pair envoie $x \oplus y$ [†]. Cela impose au second pair de télécharger y pour pouvoir décoder et

†. Dans la suite de cet article, $x \oplus y$ désignera l'opération calculant le ou-exclusif bit à bit entre x et y.



(a) Nombre de réplicats (ordonnée) par bloc (abscisse).

(b) Charge moyenne du serveur

FIGURE 2: Comparaison des variantes de *Phosphite* avec un système sans *Phosphite*

utiliser x. La procédure se répète récursivement puisque, quand le pair émet une requête pour y, il peut recevoir $y \oplus z$ avec une probabilité $p_e(y)$. La figure 1 présente un échange entre deux pairs.

Ce même challenge peut aussi être construit à partir du chiffrement symétrique de Vernam qui permet de chiffrer x avec une clé k au moins aussi longue que x en calculant $x \oplus k$. Avant de transmettre des données, x est chiffré avec la clé y. Le pair qui reçoit $x \oplus y$ doit télécharger y avant de pouvoir déchiffrer x.

Phosphite s'appuie sur deux paramètres $p_e(x)$, qui est la probabilité d'envoyer un bloc encodé, et $p_c(x,y)$, qui définit comment choisir y quand un pair a décidé d'encoder. Ces deux paramètres définissent entièrement le mécanisme d'incitation. Ainsi, comme le pair qui envoie une requête ne peut pas mentir sur le bloc x qu'il souhaite et comme *Phosphite* ne s'appuie que sur cette information, le mécanisme d'incitation laisse peu de place aux attaques.

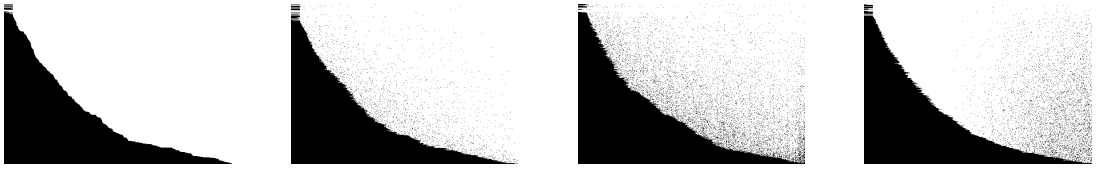
La probabilité d'encoder $p_e(x)$ peut prendre une forme quelconque. Cependant, nous nous limiterons au cas où $p_e(x) = c$ est constant puisque cette forme simple donne d'excellents résultats alors que des formes plus compliquées (croissantes, décroissantes, triangulaires) présentent des effets indésirables. En ce qui concerne la loi $p_c(x,y)$ définissant le choix du bloc avec lequel encoder, deux alternatives sont intéressantes. D'une part, le bloc y peut être choisi aléatoirement et uniformément parmi les blocs $z > x$ disponibles. D'autre part, le bloc y peut-être choisi comme étant le bloc $z > x$ disponible le plus rare localement (estimation réalisée selon les cartes de contenus des voisins du pair qui répond à la requête). Cette méthode, inspirée de la politique *LRF* de *Bittorrent*, permet d'éviter l'apparition de blocs rares. Puisque $y > x$, le pair acquiert obligatoirement une nouvelle donnée à chaque bloc reçu. Ainsi, il n'y a pas de surcoût en terme de bande passante consommée.

Puisque le mécanisme n'offre pas d'avantages aux nouveaux arrivants et aux pairs qui déclarent coopérer entre eux, les attaques classiques (attaque sibylline et collusion) ne présentent pas d'intérêt. Néanmoins, le mécanisme d'incitation reste sensible à des attaques où un pair tente de déterminer si le contenu est encodé pour fermer la connexion et annuler le téléchargement si c'est le cas.

Afin de prévenir ce type d'attaque, les entêtes du paquet doivent être ajoutés à la fin du paquet et non au début. De plus, les données doivent être chiffré symétriquement, avec une clé aléatoire qui sera ajoutée à la fin du paquet. En effet, si le contenu n'est pas chiffré, un échantillon des données permet à un pair de se faire une idée de la nature du contenu. Si un pair télécharge la même donnée depuis trois pairs et qu'au moins deux d'entre eux lui envoient un contenu dont les premiers octets sont identiques, alors il sait que ces pairs envoient du contenu non-encodé. Il choisira donc de télécharger depuis l'un de ces deux pairs. Pour prévenir cette attaque, le contenu ne doit être exploitable qu'une fois que l'intégralité du bloc a été téléchargée.

3 Évaluation

Nous effectuons une simulation par cycle d'un système où les pairs sont indépendants. Le système est homogène : tous les pairs sont identiques. Les pairs téléchargent dans l'ordre les différents blocs d'un film



(a) $p_e = 0$ (b) $p_e = 0.10$, (Uniforme) (c) $p_e = 0.25$, (Uniforme) (d) $p_e = 0.10$, (LRF)

FIGURE 3: Disponibilité des blocs suivant les pairs. Chaque ligne représente un bloc et chaque colonne représente un pair. Si un bloc est disponible, un point noir apparaît.

en anticipant jusqu'à 20 blocs par rapport à la lecture. Après une période de pré-chargement de 60 cycles, le système commence à lire les 512 blocs. Les 20 derniers blocs correspondent au générique du film. Les utilisateurs restent regarder le film qu'ils ont acheté mais quittent le système rapidement à l'issue du film. Pendant le film, ils partent selon une loi exponentielle, en moyenne au bout de 200 cycles. Pendant le générique, ils partent en moyenne au bout de 10 cycles. Une fois la lecture du film terminée, ils partent en moyenne au bout de 2 cycles.

La figure 2(b) présente le nombre moyen de blocs fournit par le serveur par cycle. L'utilisation du mécanisme d'incitation utilisant $p_e = 0.1$ et une politique *LRF* permet de diviser par 4 la charge du serveur. La diffusion entre pairs est plus efficace.

La figure 2(a) présente le nombre de réplicats pour chaque bloc. Les lignes horizontales indiquent des bornes à ne pas franchir pour que le contenu soit disponible sur au moins un voisin avec une certaine probabilité. Le mécanisme d'incitation, avec la politique *LRF*, assure que les différents blocs restent disponibles avec plus de 98% de chances alors qu'en l'absence d'incitation, les derniers blocs ne sont même pas disponibles à 50%. Ainsi, le mécanisme d'incitation permet de réduire la dépendance au serveur.

La figure 3 affiche une carte du contenu disponible sur chaque pair. Conformément aux figures 2(b) et 2(a), la politique *LRF* permet de placer judicieusement des réplicats supplémentaires sur les derniers blocs. La politique uniforme est moins efficace et nécessite de forcer plus souvent les pairs à télécharger pour obtenir une réplication aussi efficace.

4 Conclusions

Phosphite est un mécanisme d'incitation au téléchargement dans le désordre à la fois simple et puissant. Il ne présente pas de failles permettant de le contourner et reste simple à mettre en oeuvre. Sa simplicité lui permet de s'intégrer facilement dans des systèmes existants et de s'interfacer simplement avec des mécanisme de réciprocation (assurant la participation).

Phosphite s'appuie sur la construction de codes correcteur d'erreurs (ou sur le chiffrement de Vernam) afin de construire un challenge informatique qui permet d'assurer que les pairs téléchargent une partie des blocs dans le désordre.

Phosphite permet ainsi d'assurer que tous les blocs sont disponibles avec plus de 98% de chances quand l'absence de mécanisme d'incitation conduit à ne même pas pouvoir garantir 50% de chances pour les derniers blocs.

Références

[CE07] Niklas Carlsson and Derek L. Eager. Peer-assisted on demand streaming of stored media using bittorrent-like protocols. In *IFIP/Networking*, 2007.

[Coh03] Bram Cohen. Incentives build robustness in bittorrent. In *P2PEcon*, June 2003.

[CSDP07] Yung Ryn Choe, Derek L. Schuff, Jagadeesh M. Dyaberi, and Vijay S. Pai. Improving vod server efficiency with bittorrent. In *MM*, 2007.

[PWMC08] Khandoker Parvez, Carey Williamson, Anirban Mahanti, and Niklas Carlsson. Analysis of bittorrent-like protocols for on-demand stored media streaming. In *SIGMETRICS*, 2008.